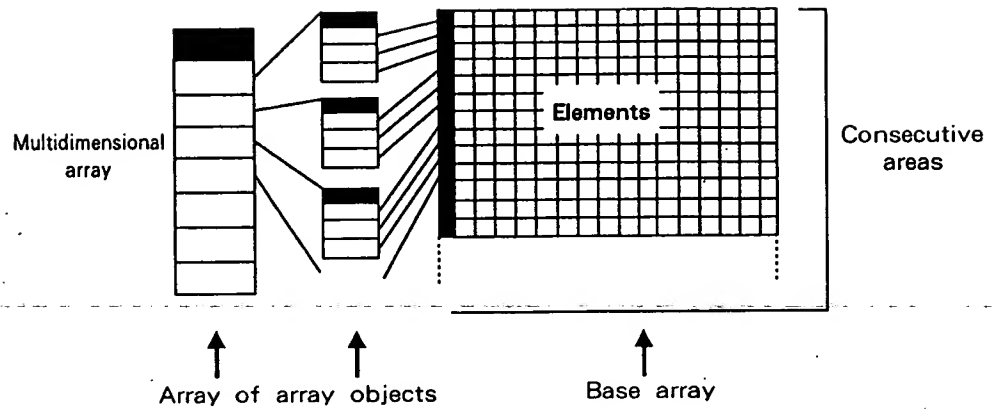

Fig. 1

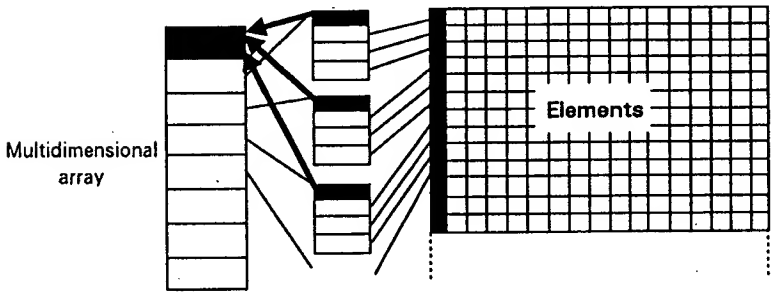

Fig. 2

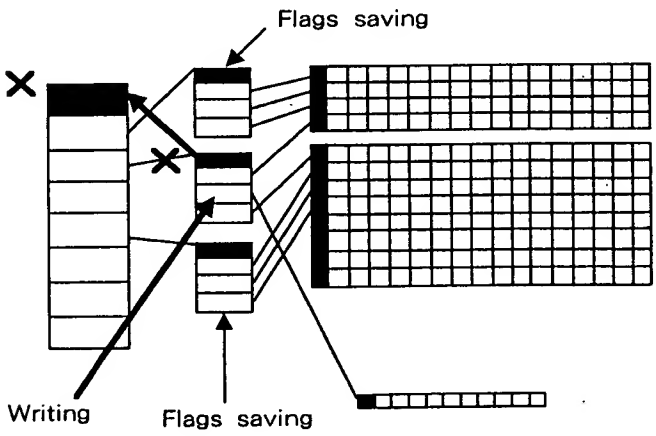

Fig. 3

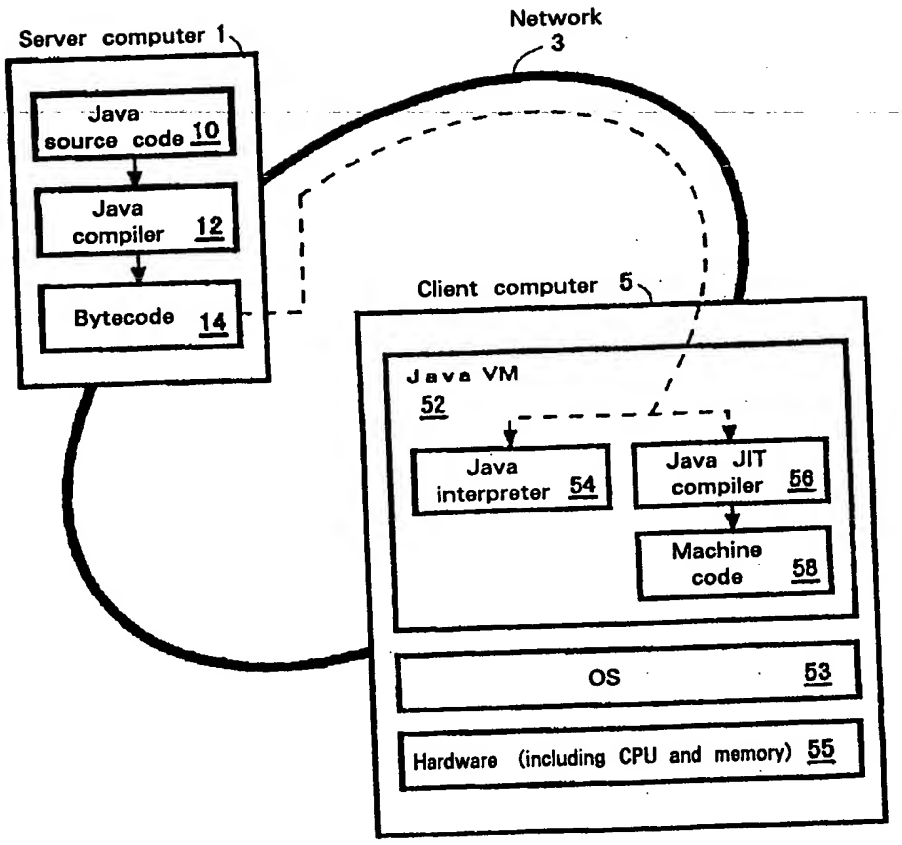
Server computer 1
Network
3
Java source code 10
Java compiler 12
Bytecode 14
Client computer 5
Java VM
52
Java interpreter 54
Java JIT compiler 56
Machine code 58
OS 53
Hardware (including CPU and memory) 55

Fig. 4

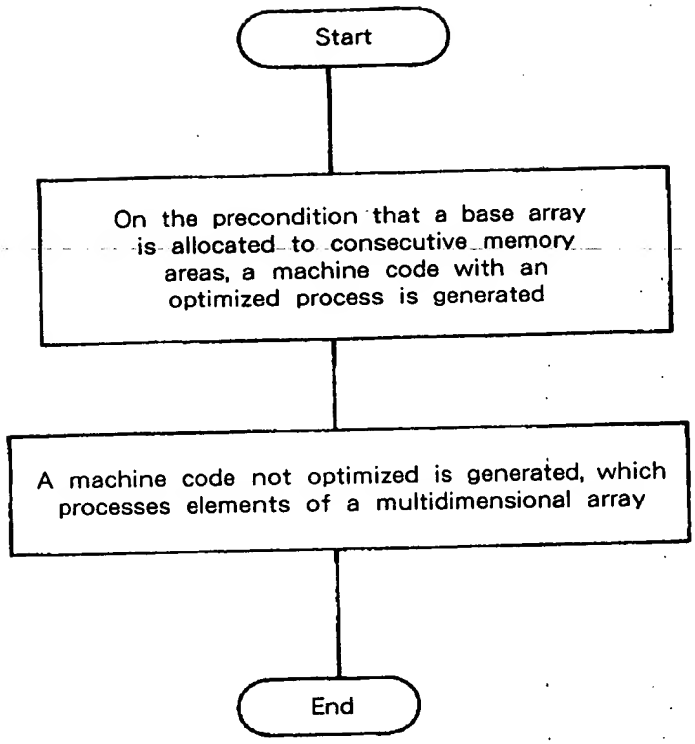

Fig. 5

0 5 6 0 5 8 0 : 0 1 2 0 5 0 0

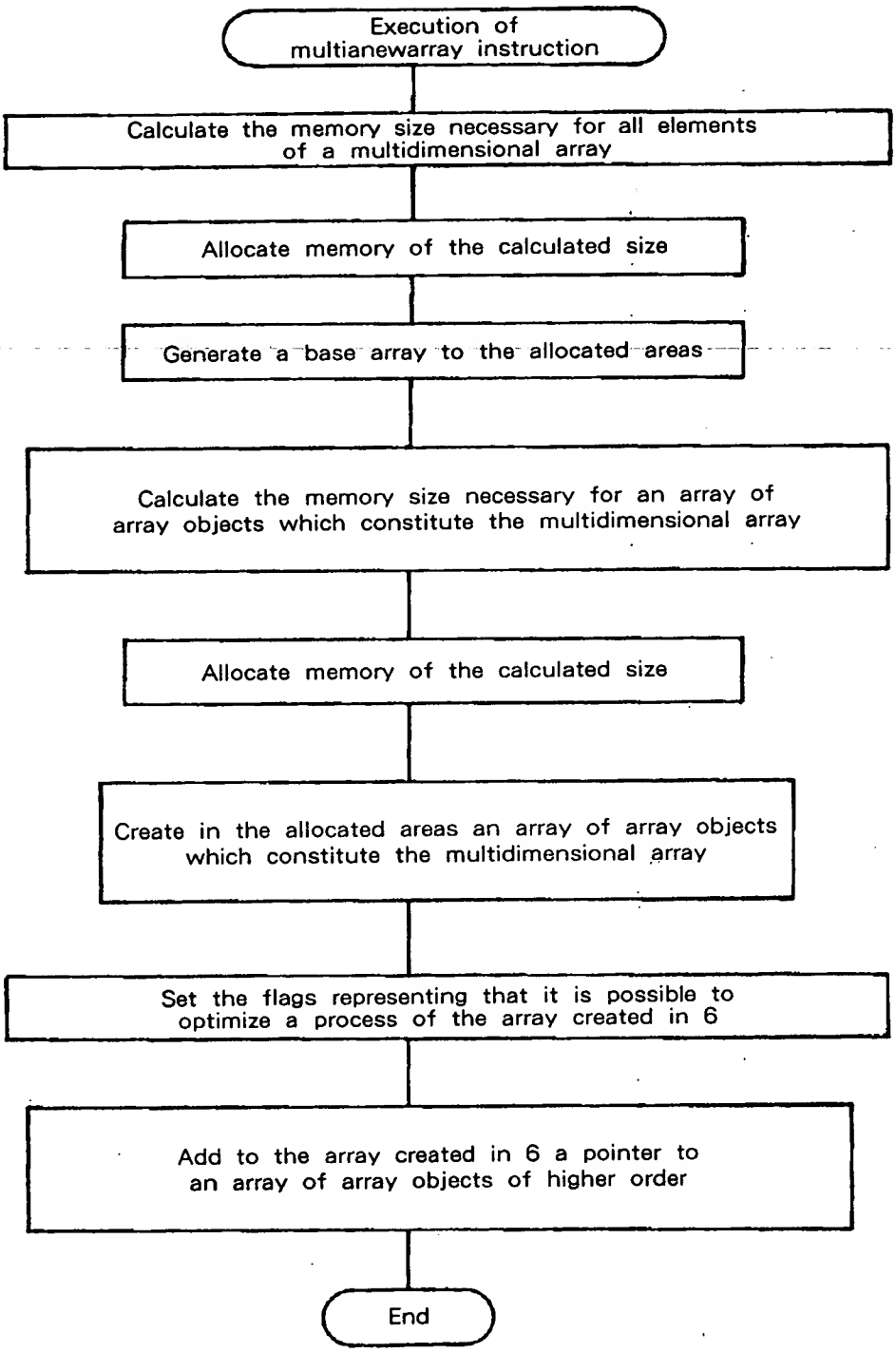
Execution of multianewarray instruction
Calculate the memory size necessary for all elements of a multidimensional array
Allocate memory of the calculated size
Generate a base array to the allocated areas
Calculate the memory size necessary for an array of array objects which constitute the multidimensional array
Allocate memory of the calculated size
Create in the allocated areas an array of array objects which constitute the multidimensional array
Set the flags representing that it is possible to optimize a process of the array created in 6
Add to the array created in 6 a pointer to an array of array objects of higher order
End

Fig. 6

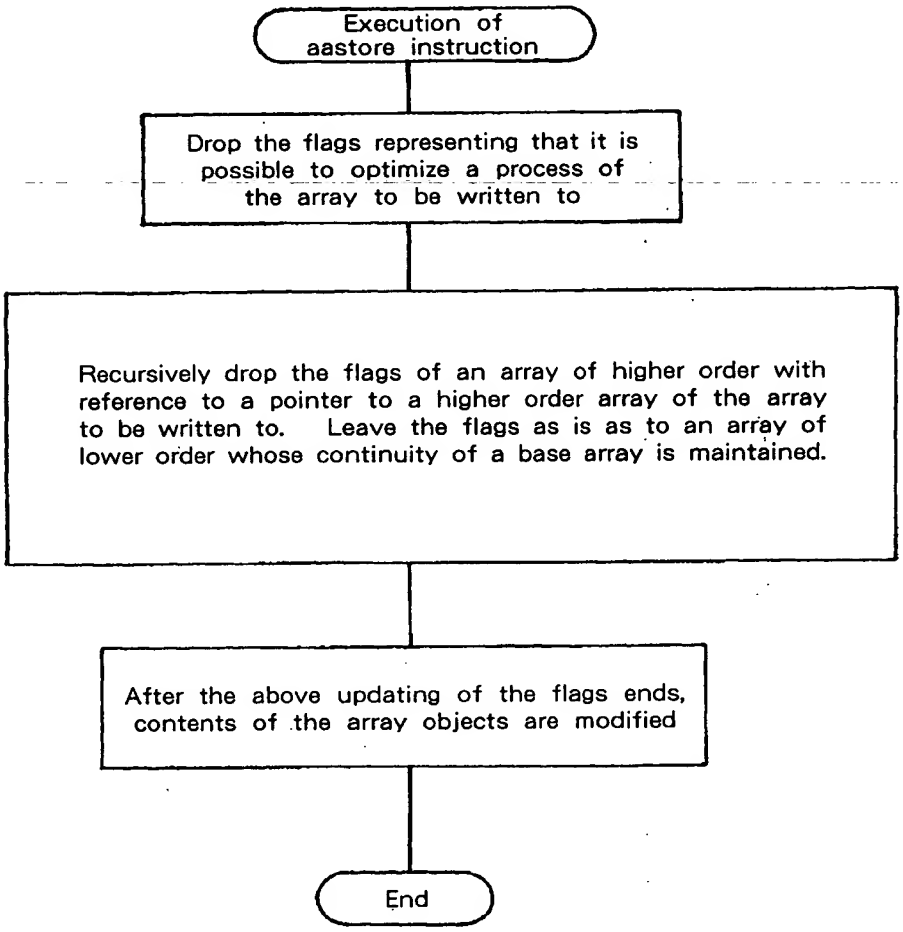
Execution of aastore instruction
Drop the flags representing that it is possible to optimize a process of the array to be written to
Recursively drop the flags of an array of higher order with reference to a pointer to a higher order array of the array to be written to. Leave the flags as is as to an array of lower order whose continuity of a base array is maintained.
After the above updating of the flags ends, contents of the array objects are modified
End

Fig. 7

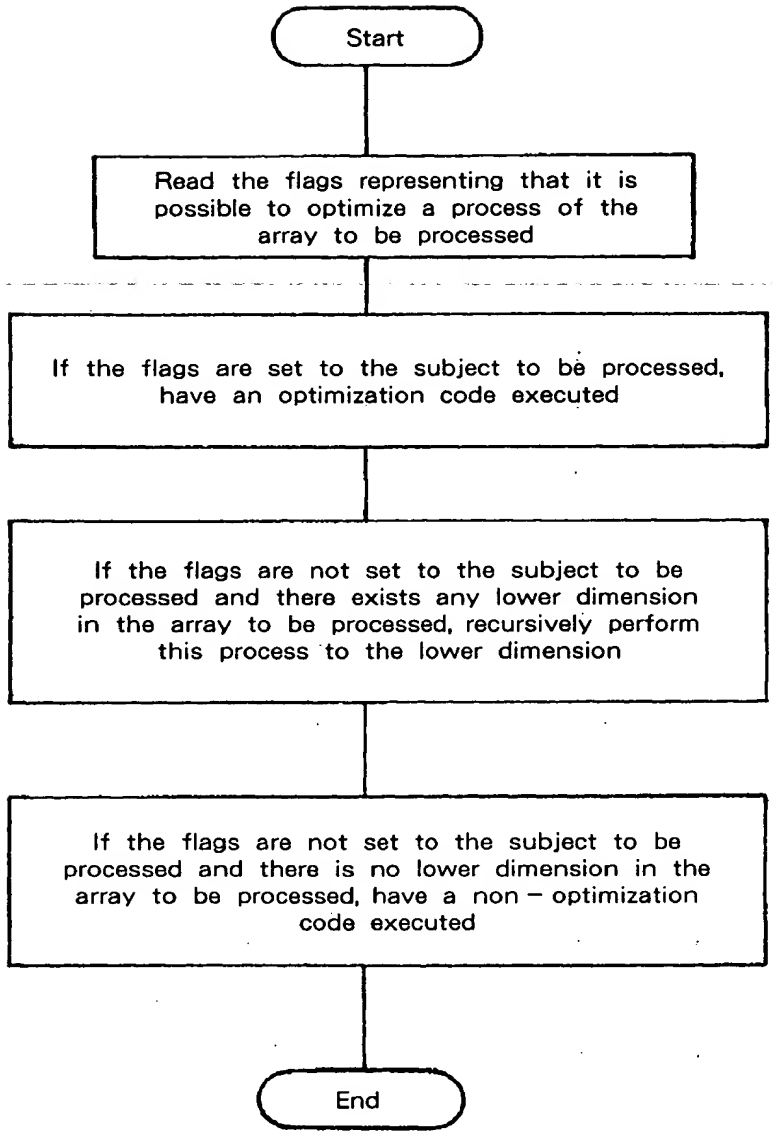
Start
Read the flags representing that it is possible to optimize a process of the array to be processed
If the flags are set to the subject to be processed, have an optimization code executed
If the flags are not set to the subject to be processed and there exists any lower dimension in the array to be processed, recursively perform this process to the lower dimension
If the flags are not set to the subject to be processed and there is no lower dimension in the array to be processed, have a non – optimization code executed
End

Fig. 8